

Efficient Splitting-based Method for Global Image Smoothing

Youngjung Kim¹, Dongbo Min², Bumsub Ham³, and Kwanghoon Sohn¹

¹Yonsei University, ²Chungnam National University, ³Inria

Abstract. Edge-preserving smoothing (EPS) can be formulated as minimizing an objective function that consists of data and prior terms. This global EPS approach shows better smoothing performance than a local one that typically has a form of weighted averaging, at the price of high computational cost. In this paper, we introduce a highly efficient splitting-based method for global EPS that minimizes the objective function of l_2 data and prior terms (possibly non-smooth and non-convex) in linear time. Different from previous splitting-based methods that require solving a large linear system, our approach solves an equivalent constrained optimization problem, resulting in a sequence of 1D sub-problems. This enables linear time solvers for weighted-least squares and -total variation problems. Our solver converges quickly, and its runtime is even comparable to state-of-the-art local EPS approaches. We also propose a family of fast iteratively re-weighted algorithms using a non-convex prior term. Experimental results demonstrate the effectiveness and flexibility of our approach in a range of computer vision and image processing tasks.

Keywords: Edge-preserving image smoothing, image filtering, weighted least squares, weighted total variation, alternating minimization, iteratively re-weighted algorithm.

1 Introduction

Edge-preserving smoothing (EPS) has attracted a strong interest in the fields of image processing and computer vision. Predominantly, it appears in a manipulation task that requires decomposing an image into a piecewise smooth layer and a detail layer. These layered signals can be recombined to match various application goals, e.g., detail enhancement and tone mapping [1]. Recent works on joint smoothing provide a new paradigm, enabling various applications such as dense correspondence [2], joint upsampling [3,4], and texture removal [5]. The problem of segmentation [6] and visual saliency [7] may also be interpreted as a joint smoothing problem. The basic idea of joint smoothing is to provide structural guidance of how the smoothing should be performed. Thus, it is assumed that the guidance signal has enough information to alter structures in the input image. In this context, global optimization-based methods [1,8,9] are advocated. They find an optimal solution to an objective function that consists of a data fidelity

term and a prior term. Thanks to such global formulation, the optimization-based methods show the state-of-the-art performance compared with local EPS approaches that typically have a form of weighted averaging [1]. However, this outperformance can be achieved only at the price of high computational cost, mainly arising from solving the global objective function. The optimization-based methods are still an order of magnitude slower than local ones [10,11,12], even with recent acceleration techniques [13,14,15]. Progress in hardware will make an efficient implementation possible, but it is not unlikely that an image resolution will increase as well. Accordingly, a different optimization technique is needed for a highly efficient global EPS methods.

In this paper, we formulate a global EPS (e.g., based on weighted-least squares or weighted-total variation) as an equivalent constrained optimization problem. This formulation results in a sequence of sub-problems that is much easier to optimize. The computational efficiency of our approach is due to a kind of variable splitting techniques. However, unlike the previous splitting-based methods¹ [14,15], our formulation enables using a highly efficient, linear time algorithm in both weighted-least squares (WLS) and -total variation (WTV) problems. As a result our approach has a time complexity linear to the number of image pixels. Another appealing aspect is that it converges with only few iterations. We also propose fast iteratively re-weighted algorithms for an objective function using a non-convex prior term. Note that the previous splitting-based methods in [14,15] are not directly applicable to many non-convex priors of practical relevance [17].

2 Problem formulation and Analysis

EPS can be formulated as finding a solution of an objective function. We can find the solution by using popular iterative methods [18] or splitting-based approaches [14,15], depending on the prior terms used in the objective function. We start with a basic formulation for EPS to provide some intuition. In the following, the subscript p denotes the 2D spatial location of a pixel.

Given an input image f and a guidance image g , a desired output u is obtained by minimizing the following objective function:

$$E(u) = \sum_p \left((u - f)_p^2 + \lambda \sum_{j \in \{1,2\}} w_{j,p} \phi(D_j u)_p \right), \quad (1)$$

where D_1 and D_2 are discrete implementations of the derivative in horizontal and vertical directions, respectively. λ controls the strength of the smoothing. g can be the input image f or a different signal correlated with f . The weight $w_{j,p} = \exp(-(D_j g)_p^2 / \kappa)$ is defined using g and constant κ . The potential function

¹ The most expensive operation in previous methods, in the senses of alternating minimization [14,15] or half quadratic optimization [16], is fast Fourier transforms with $O(n \log n)$ complexity.

ϕ and the weight function w allow one to employ various image priors that behave differently in preserving or smoothing image features. We always assume that f and g have the same width (W) and the height (H).

2.1 WLS for EPS

When $\phi = \tau^2$, the objective function of (1) becomes quadratic, and it corresponds to the WLS framework [1]. The minimizer u satisfies the following linear system:

$$\left(\mathbf{I} + \lambda \sum_{j \in \{1,2\}} \mathbf{D}_j^T \mathbf{W}_j \mathbf{D}_j \right) \mathbf{u} = \mathbf{f}, \quad (2)$$

where \mathbf{D}_j is a discrete difference matrix, \mathbf{W}_j is a diagonal matrix containing the weights w_j , and \mathbf{I} is an identity matrix. Iterative solvers such as Jacobi and conjugate gradient methods [18] are applicable to solve the sparse linear system of (2). Since these methods consist of matrix-vector multiplications, each iteration runs in linear time to an image size. However, the number of iterations, required for achieving a particular accuracy, depends on the matrix dimension ($HW \times HW$) [13], and hence the computational cost is considerable. Despite recent progress in preconditioning techniques, all the existing solvers are still an order of magnitude slower than the state-of-the-art local EPS approaches [10,11]. Moreover, the cost of constructing the preconditioner may outweigh the speed gain from the improved conditioning.

Similarly, the preconditioning techniques [13] may not accelerate EPS algorithms using the iteratively re-weighted least squares (IRLS) method [19]. An intermediate linear system of the IRLS method varies during *external* iterations and thus a series of preconditioners should be constructed, leading to a huge amount of computational overhead.

2.2 WTV for EPS

The WTV prior, $\phi = |\tau|$, often achieves a better capability along boundaries [5,8], but it needs more computational cost than using $\phi = \tau^2$ in the WLS. A common approach to minimizing the WTV objective function is to exploit the variable-splitting and penalty techniques, as follows [14]:

$$E(u, v, \beta) = \sum_p \left((u - f)_p^2 + \sum_{j \in \{1,2\}} \left(\frac{\beta}{2} (v_j - D_j u)_p^2 + \lambda w_{j,p} |v_j|_p \right) \right), \quad (3)$$

where β is a penalty parameter. An auxiliary variables v_1 and v_2 are introduced for an alternative minimization of the data and the prior terms. When $v = (v_1, v_2)$ is fixed, minimizing the objective function of (3) with respect to u can be solved in a closed-form, and vice versa. The solver is hence iteratively applied while updating u , v , and β :

$$\begin{aligned}
u^{t+1} &= \arg \min E(u, v^t, \beta^t), \\
v^{t+1} &= \arg \min_u E(u^{t+1}, v, \beta^t), \\
\beta^{t+1} &= \alpha \beta^t,
\end{aligned} \tag{4}$$

where $\alpha > 1$ is a continuation parameter. Since v^{t+1} can be obtained by *soft-thresholding* [14,15,8], the computational cost primarily lies in the u -subproblem:

$$\left(\mathbf{I} + \frac{\beta}{2} \sum_{j \in \{1,2\}} \mathbf{D}_j^T \mathbf{D}_j \right) \mathbf{u}^{t+1} = \frac{\beta}{2} \sum_{j \in \{1,2\}} \mathbf{D}_j^T \mathbf{v}_j^t + \mathbf{f}. \tag{5}$$

Although numerous methods, such as alternating direction method of multipliers (ADMM) [15] and split-Bregman (SB) [8], have been proposed, they use the fast Fourier Transform² (FFT) to update the variable u . As a result, the computational cost required for solving (3) is $O(n \log n)$ per iteration ($n = HW$).

3 Method

In this section, we first propose an efficient method of minimizing (1) when $\phi = \tau^2$ or $|\tau|$. We then apply it to solve non-convex objective functions. The key idea is to decompose (1) into each spatial dimension with a proper variable splitting, and then to use a constrained optimization technique.

Consider the following optimization problem with linear equality constraint:

$$\min_{\substack{u,v \\ \text{s.t. } u=v}} \sum_p \left(\sum_{o \in \{u,v\}} \frac{1}{2} (o - f)_p^2 + \lambda \left(w_{1,p} \phi(D_1 u)_p + w_{2,p} \phi(D_2 v)_p \right) \right), \tag{6}$$

We again denote by v an auxiliary variable. In our formulation, the size of v is equal to the input f . Then, it is clear that (6) is equivalent to the original problem (1), under the constraint ($u = v$). The penalty decomposition method³ [20], associated with (6) can be written as:

$$\min_{u,v} \sum_p \left(\sum_{o \in \{u,v\}} \frac{1}{2} (o - f)_p^2 + \frac{\beta}{2} (u - v)_p^2 + \lambda \left(w_{1,p} \phi(D_1 u)_p + w_{2,p} \phi(D_2 v)_p \right) \right). \tag{7}$$

² The linear system (5) can be diagonalized by FFT. Thus, solving (5) requires three FFT calls.

³ We have tested the augmented Lagrangian method [20] to avoid using large β values (continuation), but found this method did not improve a convergence rate, while increasing a memory demand (due to Lagrange multiplier).

This problem can be solved with block coordinate descent by minimizing (7) with respects to u and v , alternately ⁴.

$$u^{t+1} = \arg \min_u \sum_p \left((u - \tilde{f})_p^2 + \frac{2\lambda}{1 + \beta^t} w_{1,p} \phi(D_1 u)_p \right), \quad (8)$$

$$v^{t+1} = \arg \min_v \sum_p \left((v - \bar{f})_p^2 + \frac{2\lambda}{1 + \beta^t} w_{2,p} \phi(D_2 v)_p \right), \quad (9)$$

where $\tilde{f} = (1 + \beta^t)^{-1}(f + \beta^t v^t)$, $\bar{f} = (1 + \beta^t)^{-1}(f + \beta^t u^{t+1})$, and $\beta^{t+1} = \alpha \beta^t$.

Now, we are ready to show the advantage of our formulation. As D_1 represents a difference operator with respect to the horizontal axis, we can decompose (8) into sub-problems defined with 1D horizontal signals only. By introducing a 1D slack variable z , we have:

$$u^{t+1,h} = \arg \min_z \sum_x \left((z - \tilde{f}^h)_x^2 + \frac{2\lambda}{1 + \beta^t} w_{1,x}^h \phi(D_1 z)_x \right), \quad (10)$$

The super-script h denotes a horizontal signal along the x dimension ($x = 1, \dots, W$). This 1D optimization process is repeated for all horizontal signals (H in number). Note that a similar result can be obtained for the auxiliary variable v . In this case, (9) is decomposed into a sub-problem with a 1D vertical signal along the y dimension ($y = 1, \dots, H$). ■

To the best of our knowledge, the variable splitting technique has been applied in such a way that $[D_1 u, D_2 u] = [v_1, v_2]$, yielding $O(n \log n)$ complexity algorithm ($n = HW$), as explained in Section 2. The previous variable splitting [15,8,16] aims to transfer $D_1 u$ and $D_2 u$ out of the potential function ϕ by introducing the auxiliary variable v_1 and v_2 , respectively. In contrast, we utilize it to decompose the original problem of (1) into a series of 1D sub-problems. This not only leads to easily solvable sub-problems, but also significantly improves the convergence rate of the algorithm (see Section 4). In the following, we present an efficient method of solving the objective form of (10) defined with a 1D horizontal signal. Its vertical counterpart can be optimized in the same manner.

3.1 1D Fast solver

WLS With $\phi = \tau^2$, (10) can be rewritten with a 1D horizontal signal \tilde{f}^h and a guide signal g^h as

$$\arg \min_z \sum_x \left((z - \tilde{f}^h)_x^2 + \tilde{w}_{1,x}^h (D_1 z)_x^2 \right), \quad (11)$$

where $\tilde{w}_{1,x}^h = (1 + \beta^t)^{-1}(2\lambda w_{1,x}^h)$. The 1D output z that minimizes the above equation is obtained by solving the following linear system of size $W \times W$.

⁴ For scalar variables, $\arg \min_e a(e - c)^2 + b(e - d)^2 = \arg \min_e (a + b)(e - \frac{ac + bd}{a + b})^2$.

$$\left(\mathbf{I} + \mathbf{D}_1^T \tilde{\mathbf{W}}_1^h \mathbf{D}_1\right) \mathbf{z}^h = \tilde{\mathbf{f}}^h. \quad (12)$$

Note that the size of \mathbf{D}_1 and \mathbf{I} is $W \times W$, not $HW \times HW$ as in (2). Interestingly, the problem (12) becomes much easier to solve than (2) since \tilde{L} is a tridiagonal matrix. We can solve this equation with $O(n)$ cost ($n = W$) by the Thomas algorithm [21]. It consists of forward-backward steps. More details can be found in the supplementary material.

WTV When $\phi = |\tau|$, (10) is written as follows:

$$\arg \min_z \sum_x \left((z - \tilde{f}^h)_x^2 + \tilde{w}_{1,x}^h |D_1 z|_x \right), \quad (13)$$

The IRLS algorithm can be applied to solve (13) approximately [19]. However, there exists a non-iterative, $O(n)$ method for the 1D total variation [22]. While this method is designed to solve 1D (un-weighted) total variation, it is possible to extend it to minimize 1D WTV. Note that this extension enables the fast iteratively re-weighted L1 (IRL1) algorithm for 2D image smoothing. See Section 3.3 for details.

We introduce the (Fenchel-Moreau) dual form of (13) as follows:

$$\min_s \sum_x (\tilde{f}_x^h - D_1^T s)_x^2, \quad \text{s.t. } |s|_x \leq \tilde{w}_{1,x}^h, \quad s_1 = s_W = 0, \quad (14)$$

where s is the dual variable. Once the solution s^* of the problem (14) is found, we can recover the solution z^* of its primal form by

$$z_x^* = \tilde{f}_x^h - s_x^* + s_{x-1}^*, \quad \text{for } 1 \leq x \leq W. \quad (15)$$

The optimality condition which characterizes the solutions z^* and s^* is then expressed as

$$\begin{cases} s_x^* = \tilde{w}_{1,x}^h & \text{if } z_{x+1}^* > z_x^* \\ s_x^* = -\tilde{w}_{1,x}^h & \text{if } z_{x+1}^* < z_x^* \\ s_x^* \in [-\tilde{w}_{1,x}^h, \tilde{w}_{1,x}^h] & \text{if } z_{x+1}^* = z_x^*. \end{cases} \quad (16)$$

More details about the derivation of (14) and (16) are available at the supplementary material.

3.2 2D Smoothing algorithm and properties

The proposed algorithm in the case of 2D image smoothing is summarized in Algorithm 1. Given an input f , a guidance g and a smoothing parameter λ , the global 1D smoothing operations are sequentially performed along with the horizontal and vertical directions. At each iteration, the input for 1D horizontal smoothing is re-calculated as the linear combination of the auxiliary variable v^t and f (line 5). In the same manner, 1D vertical smoothing is applied to the

Algorithm 1 Fast global image smoothing

```

1: procedure FAST IMAGE SMOOTHING USING  $\phi$ 
2:   Initialize  $u^{(t=1)} = v^{(t=1)} = f$ ,  $\beta^1$ ,  $\alpha$ 
3:   for  $t = 1 : T$  do
4:     for  $y = 1 : H$  do
5:        $\bar{f}^h(x) = (1 + \beta^t)^{-1}(f(x, y) + \beta^t v^t(x, y))$  for all  $(x = 1, \dots, W)$ 
6:        $\tilde{w}_1^h(x) = (1 + \beta^t)^{-1}(2\lambda w_1(x, y))$  for all  $x$ 
7:       Compute  $z$  minimizing (11) or (13) according to  $\phi$ 
8:        $u^{t+1}(x, y) = z(x)$  for all  $x$ 
9:     end for
10:    for  $x = 1 : W$  do
11:       $\bar{f}^v(y) = (1 + \beta^t)^{-1}(f(x, y) + \beta^t u^{t+1}(x, y))$  for all  $(y = 1, \dots, H)$ 
12:       $\tilde{w}_1^v(y) = (1 + \beta^t)^{-1}(2\lambda w_2(x, y))$  for all  $y$ 
13:      Compute  $z$  minimizing (11) or (13) according to  $\phi$ 
14:       $v^{t+1}(x, y) = z(y)$  for all  $y$ 
15:    end for
16:     $\beta^{t+1} = \alpha \beta^t$ 
17:  end for
18: end procedure

```

linear combination of u^{t+1} and f (line 11). These reflect the smoothing effect from the other side. The 1D smoothing parameter gradually decreases by factor of $2/(1 + \beta^t)$ (line 6, 12). The algorithm is terminated after $t = T$ iterations.

Although the original formulation (1) is decomposed into a series of sub-problems, we compute exact solutions of the objectives (8) and (9) defined on the 1D dimension. This property considerably accelerates the convergence speed of the algorithm. Moreover, since these sub-problems can be decoupled, a straightforward parallelization is possible.

Proposition 1 *As $t \rightarrow \infty$, Algorithm 1 is convergent.*

Proof See the supplementary material. ■

Our approach is not rotationally invariant, as the original formulation of (1) enforces the smoothness term to be aligned for each axis individually. For the WLS smoothing, we do not observe any visible artifacts in all experiments. A similar observation can be found in [1]. When $\phi = |\tau|$, it prefers object boundaries which are minimal in the Manhattan (L_1) distance. This may give blocky artifacts in the region boundaries. In order to alleviate this problem, Algorithm 1 can be customized to perform a smoothing using 8-neighborhood system \mathcal{N}_8 – we should introduce two more auxiliary variables (by symmetry) and additionally iterate diagonal and anti diagonal passes⁵.

Continuation parameter α Our approach uses a continuation parameter α to gradually increase a penalty parameter β , at each iteration. This ensures that

⁵ But, all results in this paper are obtained using a \mathcal{N}_4 neighborhood system.

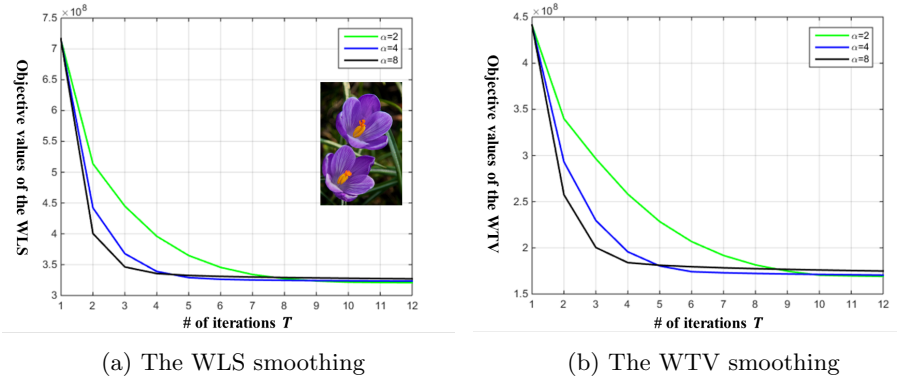


Fig. 1. The energy evolutions of our approach, depending on the continuation parameter α . (a) Our fast WLS and (b) Our fast WTV. λ and κ are set to 400 and 7.65, respectively.

the auxiliary variable v should be close to u , and thus Algorithm 1 is convergent (Proposition 1). Fig. 1 shows the energy evolutions of WLS and WTV with different values of $\alpha=2, 4$, and 8 . Using a large α makes the convergence faster, but the objective value after the convergence becomes slightly higher than that of using a small α . In this paper, we set $\alpha = 4$ by considering the trade-off between speed and accuracy.

3.3 Extension to fast iteratively re-weighted algorithms

Our approach can be extended for a wider class of potential functions by using iteratively re-weighted algorithms [23]. Let us consider general heavy-tailed potentials ψ , then the original objective function of (1) is written as

$$E(u) = \sum_p \left((u - f)_p^2 + \lambda \sum_{j \in \{1,2\}} w_{j,p} \psi(D_j u)_p \right). \quad (17)$$

The principle of iteratively re-weighted algorithms is to find a convex function, which serves as the upper bound of (17). It allows for explicit algorithms according to the construction of this bound, i.e., IRLS vs IRL1. In the following, we will use $k = (1, \dots, K)$ to represent the *external* iteration index used to minimize (17).

Fast IRLS (FIRLS) The IRLS approach uses the fact that many potentials can be seen as minima of the quadratic upper bound [23]. Using an intermediate estimate u^k , we obtain the following update rule:

$$u^{k+1} = \arg \min_u \sum_p \left((u - f)_p^2 + \lambda \sum_{j \in \{1,2\}} a_{j,p} (D_j u)_p^2 \right), \quad (18)$$

where $a_{j,p} = w_{j,p} \frac{\psi'(D_j u^k)_p}{2(D_j u^k)_p}$. Minimizing (18) is equivalent to solving a linear system (2), except that a Laplacian matrix is computed with $a_{j,p}$. Thus, we can obtain u^{k+1} using our fast WLS solver with T *internal* iterations. This extension is attractive in several aspects. First, our approach can be applied to a range of applications, which require for sparse image gradients and sharp edges (See Section 5). Second, our approach involves only simple arithmetic operations for solving (18). While most existing linear solvers [18] suffer from additionally computing the preconditioner at each *external* iteration, as an intermediate linear system varies with $k = (1, \dots, K)$.

Fast IRL1 (FIRL1) Theoretically, the IRLS algorithm can be applied only when the potential function is well approximated with a quadratic upper bound [23]. This, however, does not cover interesting functions such as $\psi = \log(1 + |\tau|)$ and L_p ($p < 1$), which are concave and non-differentiable at origin. Here we extend our fast WTV solver to the FIRL1 algorithm:

$$u^{k+1} = \arg \min_u \sum_p \left((u - f)_p^2 + \lambda \sum_{j \in \{1,2\}} b_{j,p} |D_j u|_p \right), \quad (19)$$

where $b_{j,p} = w_{j,p} \partial \psi(D_j u^k)_p$, ψ is concave, and ∂ denotes the sub-gradient. The algorithm exploits a convex function obtained by linearizing the potential ψ . Note that (19) serves as the upper bound of (17) due to concavity of ψ .

Obviously, each of the IRL1 iterations is the WTV problem, which can be solved efficiently using our solver. The pseudo-code for our fast iteratively re-weighted algorithms is provided in Algorithm 2.

Algorithm 2 Fast iteratively re-weighted algorithms

- 1: **procedure** FAST IMAGE SMOOTHING USING ψ
 - 2: Initialize $u^{(k=1)}$
 - 3: **for** $k = 1 : K$ **do**
 - 4: Construct the sub-problem (18) or (19)
 - 5: Compute u^{k+1} using Algorithm 1
 - 6: **end for**
 - 7: **end procedure**
-

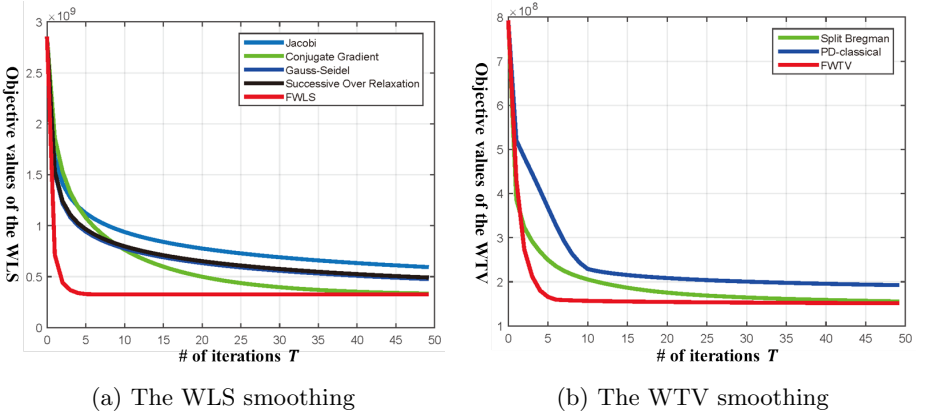


Fig. 2. Comparison of the objective decrease versus the number of iterations. (a) the WLS smoothing and (b) the WTV smoothing. Our approach rapidly reduces the objective value of (1), and converges after a few iterations (the input image is shown in the inset of Fig. 1). β^1 and α are set to 1 and 4, respectively.

4 Experimental Validation

We evaluate the convergence and runtime performance of our solver. The experiments are simulated with a single Intel i7 3.4GHz CPU. Our approach is easy to implement and we will release the source code at public website. All compared methods have been implemented in MATLAB with MEX interface. Primary parameters in our approach are set as: $\beta^1 = 1$, and $\alpha = 4$. At each iteration, we gradually increase β by factor of α . The smoothing parameters λ and κ are set to 400 and 7.65, respectively, for all methods (the range of image values is $[0 \sim 255]$).

For convergence analysis of the WLS smoothing, we compare our approach (FWLS) with the conjugate gradient (CG), Gauss-Seidel (GS), Jacobi iteration (JI), and successive over relaxation (SOR). We use the SuiteSparse library [24] to solve a triangular system, arising from GS and SOR. In the case of the WTV smoothing, our approach (FWTV) is compared to the split Bregman method (SB) [8,25] and classical penalty decomposition (classical-PD) [14]. The FFTW library [26] is used to solve the u -subproblem of (5) for SB and PD-classical methods. We show in Fig. 2(a) how the WLS objective evolves at each iteration (all methods are initialized by the input f). Although each iteration of CG, GS, JI, and SOR runs in a linear time, they require a very large number of iterations to converge. In contrast, our solver converges in a few iterations. The result of the WTV is shown in Fig. 2(b). Likewise, our approach converges much faster than SB and classical-PD. Note that these methods have $O(n \log n)$ complexity per iteration ($n = H \times W$), while our solver runs in linear time. The input image is shown in the inset Fig 1.

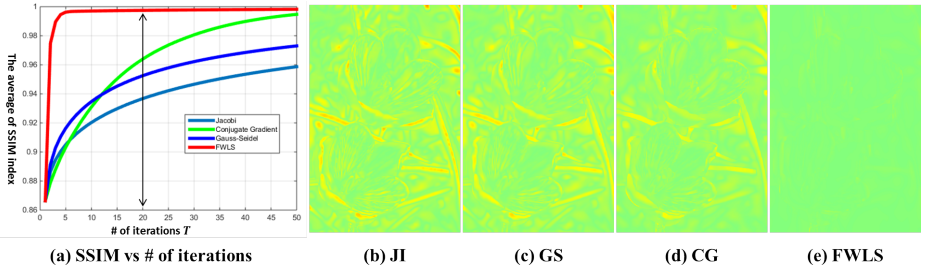


Fig. 3. The average SSIM indexes [27] for the WTV smoothing, as a function of the number of iterations. (a) The average SSIM values on 100 natural images, (b)-(d) The visualization of differences between the reference and the results of each method (at 20 iteration).

Table 1. Runtime comparison for different methods (in seconds)

Image size		PCG [13]	MATLAB “ \ ”	FWLS		SB [8,25]	FWTV
427x640	WLS	0.85	0.95	0.07	WTV	1.47	0.25
660x800		1.58	1.97	0.13		2.56	0.43
923x1128		3.04	4.14	0.28		6.69	1.02

To further demonstrate the effectiveness of our approach, we additionally collect 100 natural images from the BSDS500 dataset [28], and compare the WLS smoothing results. The smoothing result obtained by MATLAB “ \ ” command is used as the reference. The average value of the structural similarity (SSIM) index [27] is plotted as a function of the number of iterations in Fig. 3(a). The difference images between the reference and the results at 20 iteration is also visualized in Fig. 3(b)-(e). In general, we find the proposed approach yields satisfactory results after $T = 3 \sim 5$. The average SSIM values at $T = 3, 5$, and 20 are 0.9896, 0.9963, and 0.9975, respectively.

The runtime comparison is shown in Table 1. In our methods, the number of iterations T is fixed to 5 based on the above experiment. For the WLS smoothing, our approach is compared with the state-of-the-art preconditioning method (PCG) [13] and MATLAB “ \ ” operator. We note that MATLAB “ \ ” operator uses the sparse cholesky decomposition in the SuiteSparse library [24]. The result for the PCG [13] is obtained from source code provided by the author. It should be noted that although the preconditioning method [13] improves the rate of convergence significantly, constructing the preconditioner needs lots of setup time, taking about 2.6 second for 1M image. The stopping criteria of the SB [8,25] is $\|u^{k+1} - u^k\|_2 < 0.1$. Our approach is magnitude faster than competing methods.

Next, we present the analysis of our fast iteratively re-weighted algorithms. Using the FIRLS, we first minimize the objective⁶ (17) with $\psi = \sigma(1 - \exp(-\frac{\tau^2}{\sigma}))$.

⁶ The same image in Fig. 1 is used and σ is set to 7.65

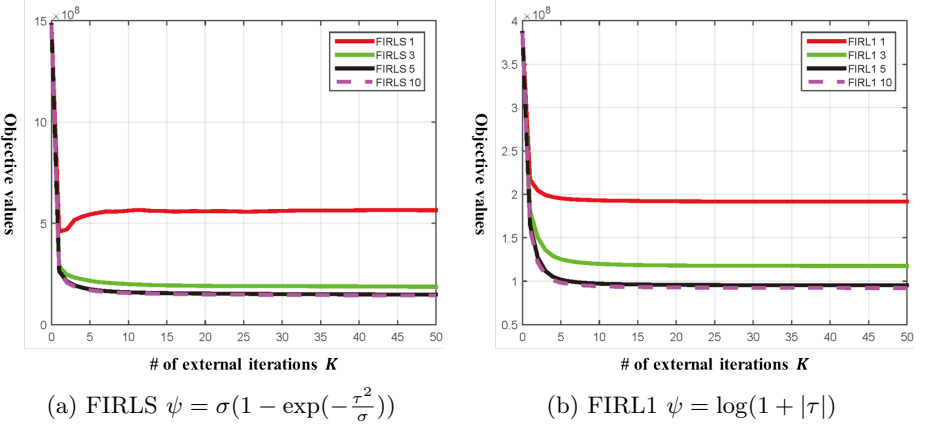


Fig. 4. Convergence of iteratively re-weighted algorithms depending on the number of inner iterations T (please see legend). (a) the FIRLS and (b) the FIRL1. Each convex surrogate functions are solved with our FWLS and FWTV solvers. If each internal iteration is not performed sufficiently, the iteratively re-weighted algorithms stuck in bad local minima.

Fig. 4(a) shows that the convergence rate of the FIRLS method differs depending on the number of inner iterations T . For the FIRL1, we use $\psi = \log(1 + |\tau|)$ as it is not well suited for the potential function that has a quadratic behavior around 0 (Fig. 4(b)). Overall, we observe that the iteratively re-weighted algorithms are stuck in bad local minima if internal iterations are not carried out sufficiently. The objective value even fluctuates with $T = 1$, as the red line of Fig. 4(a). It is thus crucial to solve the subproblems of iteratively re-weighted algorithms until reaching the certain level of accuracy. In general, we find $K = 5$, $T = 5$ is a good choice for both algorithms⁷. Many heavy-tailed potential functions ψ are non-convex, and thus any warm initializations for u^1 may further improve the convergence of our fast iteratively re-weighted algorithms.

5 Applications

Our approach is flexible, and can be applied to several tasks using EPS. In this section, we apply our method to texture removal, content-based color quantization, and style transfer. To this end, the various image priors, i.e., potential function and weight w , are exploited to match different application goals. All results and runtime in the comparison are obtained from source codes provided by the authors. The parameters are carefully tuned through extensive experiments. Additional results and other applications, such as scale-space filtering and image denoising, are also available in the supplementary material.

⁷ On this example, an average value of per-pixel difference, i.e., $\|u^{k+1} - u^k\|_2$ is 0.15 after $K = 5$ external iterations (the range of image values is $[0 \sim 255]$).

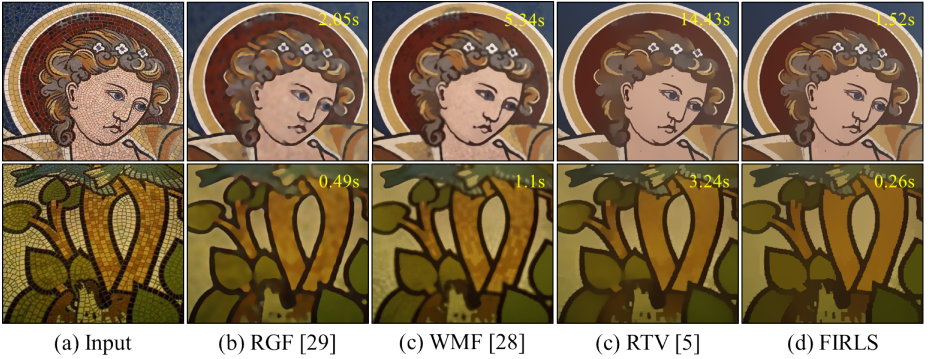


Fig. 5. Examples of the texture removal. For each method, the running times are reported in seconds (yellow). The image sizes are 1254×1067 (top) and 495×536 (bottom), respectively.

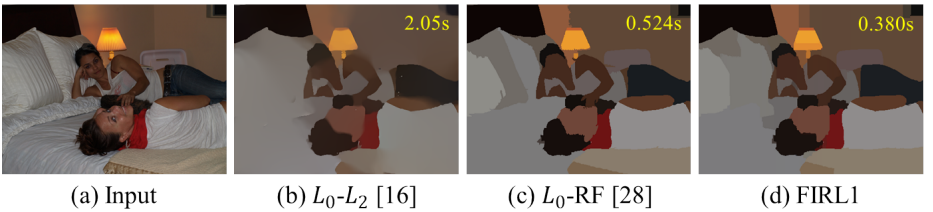


Fig. 6. Examples of the content-based color quantization. Our approach shows performance comparable to state-of-the-art L_0 minimization [29]. The running time is also reported in seconds (the image sizes are 494×371).

5.1 Texture removal

For texture removal, we employ the model proposed in [9]: ψ is set to $\sigma(1 - \exp(-\tau^2/\sigma))$ and $g = G * f$ where G is the Gaussian kernel with standard deviation 2. This type of guidance image is very effective since textures on the object are usually of small scale structures. The smoothing parameters κ , and σ are fixed to 5 and 7.65, respectively, but λ varies according to images. In this setting, we minimize the objective (17) using our FIRLS solver. Fig. 5 shows examples of texture removal obtained by the rolling guidance filter (RGF) [31], the weighted median filter (WMF) [30], the relative total variation (RTV) [5], and our FIRLS. The RGF [31] is implemented using fast bilateral filter [12]. The proposed method is even faster than the texture smoothing tools based on the fast local filtering (RGF [31], WMF [30]), while outperforming them in the subjective evaluation. Note that minimizing the objective function in the RTV [5] needs to solve a large linear system iteratively⁸. Thus, it can be also accelerated by our FWLS solver.

⁸ In the RTV [5], the prior term is defined in a form of total variation, resulting in a nonlinear system of equation. It can be easily addressed by using a fixed point iteration.



Fig. 7. Style transfer effect. The edge map of the input is used as the guidance image. The image size is 640×480 . The input image (a) is taken from [30].

5.2 Color quantization

(Content-based) color quantization attempts to increase the sparsity of colors while maintaining the overall structure in images. This is useful for many vision tasks, including image retrieval and segmentation. We use the sparsity prior, $\psi = \log(1 + |\tau|)$, to reduce the number of colors. The guidance image is not used in this application, i.e., $w = 1$. With this, we minimize the objective (17) using our FIRL1 solver. Results and comparisons are provided in Fig. 6. Our solver shows performance comparable to state-of-the-art L_0 minimization [16, 29], while running faster. The result in Fig. 6(c) has been obtained using the region fusion (RF) approach [29], which is tailored to L_0 minimization only.

5.3 Style transfer

Any feature of input image or others can be taken as the guidance g . To transfer structure of g to input images f , we use our FWLS and FWTV solvers. Fig. 7 shows a specific style transfer example obtained by the weighted median filter (WMF) [30] and our FWLS. The edge map of f is used as the guidance image g . The window size of the WMF [30] is set to 5×5 , taking about 0.4 seconds. Our FWLS and FWTV solvers take only 0.08 and 0.28 seconds, respectively, for an image of size 640×480 .

6 Conclusions

We introduce a highly efficient splitting-based method for global EPS. Unlike previous splitting-based methods, our formulation enables linear time solvers for WLS and WTV problems. Our solver converges quickly, and its runtime is comparable to state-of-the-art local EPS approaches. We also propose fast iteratively re-weighted algorithms for a non-convex objective function. Our approach is flexible, and thus is applicable to a variety of applications.

7 Acknowledgements

References

1. Z. Farbman, R. Fattal, D.L., Szeliski, R.: Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM TOG* **27**(3) (2008)
2. C. Rhemann, A. Hosni, M.B.C.R., Gelautz, M.: Fast cost-volume filtering for visual correspondence and beyond. *in Proc. CVPR* (2008)
3. J. Kopf, M. Cohen, D.L., Uyttendaele, M.: Joint bilateral upsampling. *ACM TOG* **26**(3) (2007)
4. J. Park, H. Kim, Y.W.T.M.S.B., Kweon, I.: High quality depth map upsampling for 3d-tof cameras. *in Proc. ICCV* (2011)
5. L. Xu, Q. Yan, Y.X., Jia, J.: Structure extraction from texture via relative total variation. *ACM TOG* **31**(6) (2012)
6. T. Kim, K.L., Lee, S.: Generative image segmentation using random walks with restart. *in Proc. ECCV* (2008)
7. F. Perazzi, P. Krähenbühl, Y.P., Hornung, A.: Saliency filters: Contrast based filtering for salient region detection. *in Proc. CVPR* (2012)
8. S. Bi, X.H., Yu, Y.: An l_1 image transform for edge-preserving smoothing and scene-level intrinsic decomposition. *ACM TOG* **34**(4) (2015)
9. B. Ham, M.C., Ponce, J.: Robust image filtering using joint static and dynamic guidance. *in Proc. CVPR* (2015)
10. K. He, J.S., Tang, X.: Guided image filtering. *in Proc. ECCV* (2010)
11. Gastal, E.S.L., Oliveira, M.M.: Domain transform for edge-aware image and video processing. *ACM TOG* **30**(4) (2013)
12. J. Chen, S. Paris, F.D.: Real-time edge-aware image processing with the bilateral grid. *ACM TOG* **26**(3) (2007)
13. D. Krishnan, R.F., Szeliski, R.: Efficient preconditioning of laplacian matrices for computer graphics. *ACM TOG* **32**(4) (2013)
14. Y. Wang, J. Yang, W.Y., Zang, Y.: A new alternating minimization algorithm for total variation image reconstruction. *SIAM* **1**(3) (2008)
15. M. V. Afonso, J.M.B.D., Figueiredo, M.A.T.: Fast image recovery using variable splitting and constrained optimization. *IEEE TIP* **19**(9) (2010)
16. L. Xu, C. Lu, Y.X., Jia, J.: Image smoothing via l_0 gradient minimization. *ACM TOG* **30**(6) (2011)
17. Schmidt, U., Roth, S.: Shrinkage fields for effective image restoration. *in Proc. CVPR* (2014)
18. Saad, Y.: Iterative methods for sparse linear system. *SIAM*, Philadelphia (2003)
19. Chartrand, R., Yin, W.: Iteratively reweighted algorithms for compressive sensing. *in Proc. ICASSP* (2008)
20. Nocedal, J., Wright, S.: Numerical optimization. *Springer* (2006)
21. Golub, G.H., Loan, C.F.V.: Matrix computations. *JHU Press* (2006)
22. Condat, L.: A direct algorithm for 1d total variation denoising. *IEEE SPL* **20**(11) (2013)
23. P. Ochs, A. Dosovitskiy, T.B., Pock, T.: On iteratively reweighted algorithms for nonsmooth nonconvex optimization in computer vision. *SIAM* **8**(1) (2015)
24. Online.: <http://faculty.cse.tamu.edu/davis/suitesparse.html>.
25. Goldstein, T., Osher, S.J.: The split bregman method for l_1 -regularized problems. *JIS* **2**(2) (2009)
26. Online.: <http://www.fftw.org/>.
27. Z. Wang, A. C. Bovik, H.R.S., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. *IEEE TIP* **13**(4) (2004)

28. P. Arbelaez, M. Maire, C.F., Malik, J.: Contour detection and hierarchical image segmentation. *IEEE TPAMI* **33**(5) (2011)
29. Nguyen, R.M.H., Brown, M.S.: Fast and effective l_0 gradient minimization by region fusion. *in Proc. ICCV* (2015)
30. Q. Zhang, L.X., Jia, J.: 100+ times faster weighted median filter. *in Proc. CVPR* (2014)
31. Q. Zhang, X. Shen, L.X., Jia, J.: Rolling guidance filter. *in Proc. ECCV* (2014)